A Study on Multi-label Classification

Clifford A. Tawiah and Victor S. Sheng

Department of Computer Science, University of Central Arkansas,
Conway, Arkansas, USA 72035
{ctawiah2,ssheng}@uca.edu

Abstract. Multi-label classifications exist in many real world applications. This paper empirically studies the performance of a variety of multi-label classification algorithms. Some of them are developed based on problem transformation. Some of them are developed based on adaption. Our experimental results show that the adaptive Multi-Label K-Nearest Neighbor performs the best, followed by Random k-Label Set, followed by Classifier Chain and Binary Relevance. Adaboost.MH performs the worst, followed by Pruned Problem Transformation. Our experimental results also provide us the confidence of existing correlations among multi-labels. These insights shed light for future research directions on multi-label classifications.

Keywords: multi-label classification, Multi-Label K-Nearest Neighbor, Random k-Label Set, Adaboost.MH, Classifier Chain, Binary Relevance, Pruned Problem Transformation.

1 Introduction

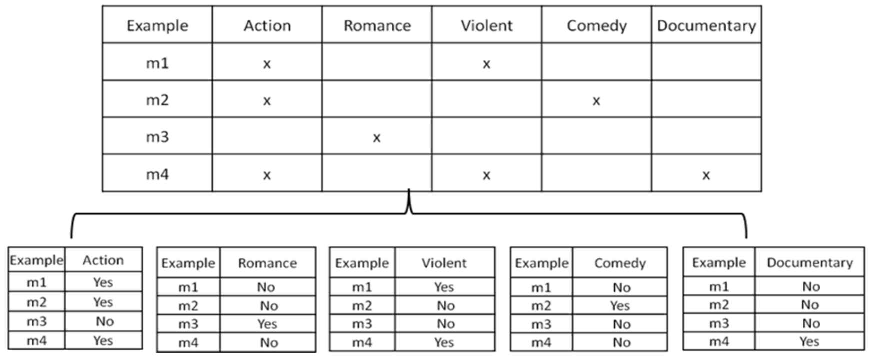
Multi-label classifications deal with multiple labels being assigned to every instance in a dataset. That is, an instance can be assigned more than one class simultaneously. It is concerned with learning a model that outputs a bipartition of a set of labels into relevant and irrelevant with respect to a query instance. This type of classification differs in some respect from traditional single label classifications in that one of multiple labels is allocated to an instance in the dataset. In single-label classifications each instance is associated with a single label and a classifier learns to associate each new test instance with one of these known labels [1, 4].

Multi-label classification tasks exist in many real-world applications, such as, gene classification in bioinformatics [8], medical diagnosis, document classification, music annotation, image recognition, and so on. All these applications require effective and efficient multi-label classification algorithms. There exist a variety of multi-label classification algorithms [10]. For data mining practitioners, it is very important for them to know the general knowledge on which algorithms perform the best, such that they can try to apply it first. For data mining researchers, it is important to investigate the performance of the existing algorithms to get insights, such that they can use the clue to guide their research on developing more effective multi-label classification algorithms.

Current existing multi-label classification algorithms are developed based on two basic approaches: algorithm adaptation and problem transformation. Problem transformation is easy to understand. We discuss it first.

Before we discuss the procedure of problem transformation, let us review traditional classifications a bit. On the contrast, traditional classifications can be called single label classifications. Giving a set L of labels, traditional single label classifications choose one label from the set to assign to an instance. If $|L| = 2$, then the problem is binary classification. Otherwise, if $|L| > 2$, it becomes a multi-class classification problem. On the contrary, multi-label learning is to assign multiple different labels to a test instance simultaneously [9].

Problem transformation is to transfer multi-label classifications into multiple traditional single label classifications, specifically, multiple binary classifications. Figure 1 shows an example of the process of transferring a multi-label classification (movie classification) into multiple binary classifications (yes or no).



Example	Action	Romance	Violent	Comedy	Documentary
m1	x		x		
m2	x			x	
m3		x			
m4	x		x		x

Example	Action
m1	Yes
m2	Yes
m3	No
m4	Yes

Example	Romance
m1	No
m2	No
m3	Yes
m4	No

Example	Violent
m1	Yes
m2	No
m3	No
m4	Yes

Example	Comedy
m1	No
m2	Yes
m3	No
m4	No

Example	Documentary
m1	No
m2	No
m3	No
m4	Yes

Fig. 1. An example of multi-label problem transformation

After a multi-label classification problem is transferred into multiple binary classification ones. All the traditional classification algorithms can be applied directly to build a classifier for each binary dataset and make prediction for its correlated test instances. The prediction for a multi-label instance is made by aggregating outputs from autonomous binary classifiers. Binary Relevance (BR) [3, 11], Classifier Chain [3], Random *k*-Label Set [7], and Pruned Problem Transformation [12] are the examples of classifiers, which use problem transformation. We will briefly review these algorithms in the following section and make comparison among them in Section 4.

Different algorithms have their method for classifying multi-label instances. For example a classifier which employs problem transformation for its classification may apply a probability distribution over the transformed dataset to rank and assign labels to the test instances. In ranking, the task is to order labels, so that the topmost labels have a greater probability to assign to the test instance. This way the class with the highest probability will be ranked first, the class with the second best probability will be ranked second, and so on.

The second method used in multi-label classifications is algorithm adaptation. It extends existing traditional classification algorithms to perform multi-label classifications directly, for example, Adaboost.MH [15] and Multi-label K-Nearest Neighbor (MLKNN) [5]. Adaboost.MH is an adaptation of Adaboost [13] for multi-label classifications. MLKNN is an adaption of the traditional k-NN algorithm for multi-label classifications.

Algorithm adaption completely differs from problem transformation in that no binary transformation made. Instead, the algorithm learns the structure and co-relations that exist among labels to classify a test instance after being trained. Thus, it is very useful to investigate the performance of multi-label classification algorithms, which are developed based on the two approaches. The investigating results will guide data mining researchers in their future research on developing better multi-label classification algorithms.

The rest of the paper is organized as follows. Section 2 introduces the popular multi-label classification algorithms which we will make comparison empirically. Section 3 introduces five popular performance metrics specifically designed for multi-label classifications. In Section 4, we describe the experiments we have conducted. They consist of the setting of the experiments, the experimental results, and the analysis of the experimental results. Section 5 concludes with a summary of our work and a discussion of future work.

2 Popular Multi-label Classification Algorithms

We briefly review the six popular multi-label classification algorithms (i.e. Binary relevance [3, 11], Classifier Chain [3], Random K-Label Set [7], Pruned Problem Transformation [12], AdaBoost.MH [15], and Multi-label K-Nearest Neighbor [5]) in this section, which are used in our experiments in Section 4.

2.1 Binary Relevance (BR)

As we introduced before, Binary Relevance [3, 11] is one of the popular problem transformation approaches. It transfers a multi-label classification into multiple binary classifications (Figure 1). After the transformation, a traditional classification algorithm is applied to build multiple binary classifiers. Each classifier is responsible for predicting the presence or absence of each corresponding label which belongs to *L*. For example, if the total number of labels for a dataset is 70, then 70 binary classifiers would be trained for each label, with a 0 or 1 association. When classifying a new instance, this approach outputs the union of the labels that are predicted by all the binary classifiers.

2.2 Classifier Chain (CC)

Classifier Chain (CC in short) [3] is also a problem transformation method for multi-label classifications. It combines the computational efficiency of binary relevance and label dependency for classifications [3]. Classifiers are linked along a chain, where each classifier deals with the binary relevance problem associated with a label in |L|. This is how classifier chain works. When a training set is passed to the algorithm, it creates a chain of classifiers C_1, C_2, C_3, ..., $C_{|L|}$, where |L| is the total number of labels for the dataset. Each of the multi-labels of the dataset is transformed into a binary problem. Thus, each classifier (C_1, ..., $C_{|L|}$) is responsible for learning and predicting binary associations (0 or 1) for each label. If a test instance *X* is introduced to the trained model, the classification process for *X* starts from C_1 and runs down along the chain of classifiers. Each classifier determines the probability of *X* to be classified into L_1, L_2, L_3, ..., $L_{|L|}$. It sort of builds a binary tree, where each link in the chain is extended with a 0 or 1 label association. This chaining method passes label information between classifiers, allowing CC to take into account label correlations and thus overcoming the label independence problem.

2.3 Random *k*-Label Set (RA*k*EL)

RA*k*EL [7] was proposed to solve performance issues of Label Powerset (LP). LP is a simple problem transformation method, but time consuming. It produces all subsets *LP(L)* of the multi-label set *L*, and treats each subset as a label. Then, it transforms the multi-label classification into traditional single label classification. Because LP produces a great number of labels, it could cause imbalanced issues, considering the number of labels versus the number of instances. In addition, the size of the generated labels incurs considerable computational costs [7]. RA*k*EL improves it by randomly selecting *k*-sized label sets from *L*, and then performing the typical LP approach on these generated subsets.

2.4 Pruned Problem Transformation (PPT)

Pruned Problem Transformation [12] is also an improvement of LP. The only improvement is that PPT prunes away the power subsets *LP(L)* that occur fewer times than a small user-defined threshold (usually 2 or 3). Removing certain information might skew or cause information to be lost. In order to avoid this issue, PPT optionally split each of the multi-label set into subsets. Thus, the subsets could occur more than the user-defined threshold.

2.5 AdaBoost.MH (AD)

Adaboost [13] is a short form of adaptive boosting. Boosting is a meta-algorithm, meaning it can be used in conjunction with other learning algorithms for improving their performance. It combines inaccurate and rough rules to produce accurate results. Given a base learning algorithm, Adaboost works by initially setting the weights of all training instances to be equal. Then it calls the base learning algorithm several times. For each call, the weight of incorrectly classified instances is increased. This is to help the base learning algorithm focus on the misclassified instance until it is correctly classified. AdaBoost.MH [15] is an adaptation of Adaboost for multi-label classifications. Adaboost.MH works similarly like the adaboost algorithm, except that it breaks the multi-label classification down into a binary problem where each test instance is classified according to its label association (either 0 or 1).

2.6 Multi-Label *K*-Nearest Neighbor (MLKNN)

MLkNN [5] is an adaption of the traditional k-NN algorithm for multi-label classifications. It is one of lazy learning algorithms. The algorithm identifies, for each unseen test instance, the k nearest neighbors in the training set. It calculates prior probabilities from the k nearest training instances, and then finds the maximum posteriori probability to determine the label set for the test instance.

3 Performance Metrics

The performance evaluation of multi-label classification is much more complicated than traditional classification. In this Section, we introduce five popular performance metrics specifically designed for multi-label classifications [6, 15, 16], i.e., Hamming Loss, Average Precision, One-Error, Coverage, and Ranking Loss. Before describing their definitions, we explain the related notations first. Supposing there is a multi-label dataset D = $\{(xi, Yi) \mid 1 \leq i \leq p\}$, notations presented in the formulas below are: $h(xi)$ represents a set of proper labels for xi ; Δ represents symmetric difference; $h(xi, y)$ represents the value of confidence for y to be a label of xi; $rank^h(x_i, y)$ returns the rank of y from $h(xi, y)$; and $\overline{y_i}$ represents the complementary of y_i [14].

3.1 Hamming Loss (HL)

Hamming Loss takes into account prediction errors, which labels are incorrectly predicted and which labels were not predicted at all [14]. It takes into account how many instance-label pairs are misclassified. The smaller the value, the better the performance is.

$$\text{HL} = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|y|}|h(x_i)\Delta y_i| \tag{1}$$

3.2 Average Precision (AP)

Average Precision evaluates the average fraction of labels ranked above a particular label which belongs to L [14]. It is often used for the evaluation of information retrieval tasks. The bigger the value of Average Precision, the better the classification performance is.

$$\text{AP} = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|y_i|}\bullet\frac{|P_i|}{rank^h(x_i, y)}\text{, where} \tag{2}$$

$$P_i = \{y' \mid rank^h(x_i, y^i) \leq rank^h(x_i, y), y^i \in y_i\}$$

3.3 One-Error (OE)

This measure calculates the number of times that the top-ranked label predicted is not in the original label set of an instance. So it checks whether the top-ranked label is relevant, and ignores the relevancy of all other labels. The smaller the value of One- Error, the better the performance is.

$$\mathrm{OE} = \frac{1}{p}\sum_{i=1}^{p}[[\arg\max_{y\in Y} h(x_i, y)] \notin y_i] \tag{3}$$

3.4 Coverage (CV)

Coverage measures how far we need, on average, to go down the list of labels in order to cover all the possible labels assigned to an instance [14]. The goal of coverage is to assess the performance of a classifier for all the possible labels of instances. The smaller the value of Coverage, the better the performance is.

$$\mathrm{CV} = \frac{1}{p}\sum_{i=1}^{p}\max rank^{h}{}_{y\in Y_i}(x_i, y) - 1 \tag{4}$$

3.5 Ranking Loss (RL)

Ranking Loss computes the average fraction of label pairs which are not correctly ordered [14] for an instance. The smaller the value of Ranking Loss, the better the performance is.

$$\mathrm{RL} = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|Y_i||Y_i|}\bullet|R_i|\text{, where} \tag{5}$$

$$R_i = \{(y_i, y_2) \mid h(x_i, y_1) \le h(x, y_2), (y_1, y_2) \in y_i \times \overline{y_i}\}$$

4 Experiments

In this section, we will make thorough comparisons among the six multi-label classification algorithms introduced in Section 2, by applying them on eleven datasets. We evaluate their performance using the five popular metrics described in Section 3.

4.1 Experimental Setup

In our experiments, we try to conduct experiments on all available datasets listed in MULAN [10] website[1]. We have not obtained experimental results for some datasets,

[1] `http://mulan.sourceforge.net/datasets.html`

because of the limitation of our computer memory. The specifications of the computer used is a 64-bit Operating System, x64-based processor, Intel(R) Core(TM) i5-2467M with 8GB memory. We succeeded in the eleven multi-label datasets: Cal500 (human-generated musical annotations), Corel5k (learning a lexicon for a fixed image recognition), Emotions (music emotion detection), Enron (email messages) [2], Genbase (classification of protein families), Medical (variables consisting of illnesses and treatments), Scene (semantic indexing of still images), Yeast (gene function classification), Bookmarks (text tagging suggestion), Mediamill (multimedia analysis), and Bibtex (text tagging suggestion). The detail characteristics of the eleven datasets are listed in Table 1.

Table 1. Description of the datasets used in the experiments

Name	#Instances	#Training Inst.	#Test Inst.	Nominal	Numeric	Labels
Cal500	502			0	68	174
Corel5k	5000	4501	499	499	0	374
Emotions	593	391	202	0	72	6
Enron	1702	1123	579	1001	0	53
Genbase	662	463	199	1186	0	27
Medical	978	333	645	1449	0	45
Scene	2407	1211	1196	0	294	6
Yeast	2417	1500	917	0	103	14
Bookmarks	87856			2150	0	208
Mediamill	43907			0	120	101
Bibtex	7395	4880	2515	1836	0	159

Each dataset comes along with an xml header file specifying the names of the labels and hierarchical relationships among them [10]. Except Cal500, Bookmarks, and Mediamill, eight datasets in Table 1 are already separated into training and test sets. For each of these datasets, we loaded two files, an XML file, and the ARFF files for the train and test set.

We conduct experiments on the six classification algorithms described in Section 2 for each dataset. If a dataset is separated into a training and test set already, we only run each classification algorithm once on its training set, and report its performance over its test set. We have to explain how we conduct experiments on Bibtex. Because of memory limitation, we could not obtain experimental results using its original training and test set directly. Thus, we have to sample our training set (2000 instances, the maximum number of instances our computer can handle) and testing set (1200 instances) from its original training and test set respectively to conduct experiments.

There are three datasets: Cal500, Bookmarks, and Mediamill, which are not separated into train and test sets. We resample them using randomization and repeat the process ten times. The average results are presented in the paper. For Cal500, we split its whole dataset into 70% for training and 30% for testing each time. For Bookmarks and Mediamill, we sample 2000 instances as the training set and 1200 instances as the test set each time. Again, the size 2000 is the proper number of instances that our computer can process.

Notice that in our experiment the default base learner is used for the six multi-label classification algorithms. Specifically, J48 is used in conjunction with Binary Relevance and Classifier Chain. LabelPowerset, in conjunction with J48 is used as the base learner for RA*k*EL. J48 pruning tree is used for Pruned Problem Transformation. Adaboost.MH uses AdaBoostM1 as its base learner, which in turn uses decision stump as its base learner [17].

4.2 Experimental Results

Tables 2 through 9 show the experimental results of all of the six multi-label classification algorithms on eight datasets, i.e., Emotions, Enron, Genbase, Medical, Scene, Yeast, Mediamill, and Bibtex. Tables 10 through 12 show the experimental results of some of the six multi-label classification algorithms on three datasets: i.e., Cal500, Corel5k, and Bookmarks. This is because some of the algorithms run out of memory. For Cal500, we will show the experimental results for the algorithms except PPT. For Corel5k, we will show the experimental results for the algorithms except Adaboost.MH. For Bookmarks, we will show the experimental results for the algorithms except PPT and RA*k*EL. Based on the experimental results, we highlight the best in bold, highlight the second in italic, and underline the worst, for each of the five performance metrics for each dataset. Again, only for Average Precision, a higher value is better. For the rest four metrics, a lower value is better.

Table 2. Experimental results on Emotion

	HL	AP	OE	CV	RL
CC	0.2896	0.6726	0.4455	2.8267	0.3383
RAKEL	*0.2228*	*0.7841*	*0.2871*	*2.0545*	*0.1841*
AD	<u>0.3160</u>	<u>0.5906</u>	<u>0.5248</u>	<u>3.0842</u>	<u>0.4295</u>
PPT	0.3127	0.6928	0.4208	2.6832	0.3135
BR	0.2599	0.6959	0.3663	2.8069	0.3103
MLKNN	**0.2087**	**0.7965**	**0.2822**	**1.8762**	**0.1586**

Table 3. Experimental results on Enron

	HL	AP	OE	CV	RL
CC	0.0530	0.5722	0.4162	*23.2919*	*0.1794*
RAKEL	**0.0509**	*0.6051*	*0.2815*	24.7841	0.2030
AD	0.0619	<u>0.4574</u>	0.4594	27.7789	0.2370
PPT	<u>0.0727</u>	0.4703	<u>0.5043</u>	<u>27.8152</u>	<u>0.2613</u>
BR	0.0540	0.5746	0.4059	24.7997	0.1861
MLKNN	*0.0514*	**0.6345**	**0.2798**	**13.1813**	**0.0934**

Table 4. Experimental results on Genbase

	HL	AP	OE	CV	RL
CC	**0.0011**	*0.9918*	0.0050	*0.3166*	*0.0018*
RAKEL	**0.0011**	0.9900	*0.0101*	**0.2965**	**0.0016**
AD	<u>0.0456</u>	<u>0.3184</u>	<u>0.7286</u>	<u>14.0704</u>	<u>0.5281</u>
PPT	*0.0026*	0.9864	**0.0000**	0.5176	0.0063
BR	**0.0011**	*0.9918*	0.0050	*0.3166*	*0.0018*
MLKNN	0.0052	**0.9931**	**0.0000**	0.5779	0.0062

Table 5. Experimental results on Medical

	HL	AP	OE	CV	RL
CC	**0.0103**	**0.8268**	**0.1907**	4.4341	0.0756
RAKEL	0.0113	0.8126	0.2031	*4.0946*	*0.0732*
AD	<u>0.0276</u>	<u>0.3571</u>	<u>0.7395</u>	<u>13.8512</u>	<u>0.2858</u>
PPT	0.0173	0.7476	0.2713	5.9364	0.1051
BR	*0.0106*	*0.8226*	*0.1969*	4.4450	0.0763
MLKNN	0.0188	0.7266	0.3535	**3.5442**	**0.0586**

Table 6. Experimental results on Scene

	HL	AP	OE	CV	RL
CC	0.1392	0.7295	0.3712	1.3094	0.2365
RAKEL	*0.1150*	*0.8150*	*0.2977*	*0.6873*	*0.1166*
AD	<u>0.1810</u>	<u>0.4302</u>	<u>0.7860</u>	<u>2.5819</u>	<u>0.4898</u>
PPT	0.1623	0.7248	0.4130	1.1714	0.2134
BR	0.1389	0.7115	0.4264	1.2809	0.2307
MLKNN	**0.0953**	**0.8513**	**0.2425**	**0.5652**	**0.0925**

Table 7. Experimental results on Yeast

	HL	AP	OE	CV	RL
CC	0.2638	0.6295	0.3490	9.0349	0.3286
RAKEL	*0.2328*	*0.7102*	0.2901	*7.6696*	*0.2240*
AD	0.2330	<u>0.5930</u>	*0.2497*	9.2454	<u>0.3821</u>
PPT	<u>0.2947</u>	0.6470	0.3391	8.5627	0.3130
BR	0.2588	0.6164	<u>0.4024</u>	<u>9.2857</u>	0.3206
MLKNN	**0.1980**	**0.7574**	**0.2421**	**6.3642**	**0.1707**

Table 8. Experimental results (in average) on Mediamill

	HL	AP	OE	CV	RL
CC	0.0518	0.4324	0.5086	*45.1207*	*0.1661*
RAKEL	0.0481	*0.4969*	0.3147	46.0259	0.1745
AD	**0.0382**	0.4503	*0.2543*	<u>59.2112</u>	<u>0.2395</u>
PPT	0.0448	0.4222	0.3362	54.9310	0.2380
BR	<u>0.0534</u>	<u>0.3757</u>	<u>0.6336</u>	51.9397	0.2107
MLKNN	*0.0404*	**0.5682**	**0.2457**	**29.8017**	**0.1060**

Table 9. Experimental results on sampled Bibtex

	HL	AP	OE	CV	RL
CC	0.0145	**0.3784**	0.5415	*66.2824*	**0.2496**
RAKEL	**0.0136**	0.3602	**0.5183**	72.6379	0.3097
AD	0.0149	<u>0.0859</u>	<u>0.8505</u>	<u>104.302</u>	<u>0.5181</u>
PPT	<u>0.0196</u>	0.2938	0.6445	76.4153	0.3393
BR	0.0141	*0.3781*	*0.5316*	68.3189	*0.2615*
MLKNN	*0.0138*	0.2753	0.6611	**66.2425**	0.2744

Table 10. Experimental results (in average) on Cal500

	HL	AP	OE	CV	RL
CC	<u>0.1789</u>	0.3053	0.7616	169.8146	0.3746
RAKEL	0.1700	*0.3829*	0.3709	*166.3377*	*0.2983*
AD	*0.1423*	<u>0.2319</u>	**0.0927**	169.3974	<u>0.5642</u>
BR	0.1659	0.3198	<u>0.8013</u>	<u>170.1391</u>	0.3466
MLKNN	**0.1375**	**0.4916**	*0.1060*	**128.9934**	**0.1823**

Table 11. Experimental results on Corel5k

	HL	AP	OE	CV	RL
CC	0.0101	0.2392	0.7240	161.74	0.1826
RAKEL	*0.0096*	<u>0.1296</u>	0.7260	<u>333.23</u>	<u>0.6381</u>
PPT	<u>0.0163</u>	0.1767	<u>0.8040</u>	268.01	0.4332
BR	0.0098	*0.2550*	*0.7080*	*124.91*	*0.1429*
MLKNN	**0.0093**	**0.2656**	**0.7060**	**113.04**	**0.1297**

Table 12. Experimental results (in average) on bookmarks

	HL	AP	OE	CV	RL
CC	0.0112	0.2159	0.8168	84.1414	0.2834
AD	*0.0104*	<u>0.1166</u>	<u>0.9529</u>	<u>109.528</u>	<u>0.4230</u>
BR	0.0118	*0.2299*	*0.7853*	*82.5759*	*0.2818*
MLKNN	**0.0096**	**0.2423**	**0.7644**	**80.5183**	**0.2704**

In order to see clearly and to show the general knowledge of the performance of the six multi-label classification algorithms, Table 13 shows the average values of the five performance metrics over the eight datasets (Emotions, Enron, Genbase, Medical, Scene, Yeast, Mediamill, and Bibtex), from Table 2 to 9. The experimental results of other three datasets (Cal500, Corel5k, and Bookmarks) are not included in Table 13, because we did not obtain results for some of the six multi-label classification algorithms, which ran out of memory. Partial experimental results for the three datasets are shown in Tables 10 through 12.

We further summarize the comparisons among the six multi-label classification algorithms through ranking over the eight datasets. For each of the eight datasets, we ranked the performance of the six algorithms from 1 (best) to 6 (worst) on each metric. The average rank of the six algorithms on each metric is shown in Table 14. Further, we average the average ranks for each of the six algorithms across the five performance metrics in the last column of Table 14. Again, the experimental results of other three datasets (Cal500, Corel5k, and Bookmarks) are not included.

In Table 13 and 14, we also highlight the best in bold, highlight the second in italic, and underline the worst for each of the five performance metrics and the overall average ranks (Table 14 only) for the six algorithms.

Table 13. Average Results of the eight datasets for different algorithms

	HL	AP	OE	CV	RL
CC	0.1029	0.6541	0.3534	*19.077*	0.1969
RAKEL	*0.0869*	*0.6967*	**0.2753**	19.781	*0.1608*
AD	0.1147	<u>0.4103</u>	<u>0.5741</u>	<u>29.265</u>	<u>0.3887</u>
PPT	<u>0.1158</u>	0.6231	0.3661	22.254	0.2237
BR	0.0988	0.6458	0.3710	20.399	0.1997
MLKNN	**0.0789**	**0.7003**	*0.2883*	**15.269**	**0.1200**

Table 14. Average ranks of different algorithms on the eight datasets

	HL	AP	OE	CV	RL	**Average**
CC	3.375	3.00	3.5	3.125	3.125	3.225
RAKEL	**2.0**	*2.625*	*2.375*	*2.375*	*2.5*	*2.375*
AD	4.5	<u>4.5</u>	<u>4.625</u>	<u>5.75</u>	<u>5.75</u>	<u>5.025</u>
PPT	<u>4.75</u>	4.125	3.625	4.375	4.375	4.25
BR	3.25	3.5	3.625	4.0	3.25	3.525
MLKNN	**2.0**	**2.0**	**2.0**	**1.5**	**1.5**	**1.8**

Table 14 shows that MLKNN performs the best. Its average ranks on all five performance metrics are the best (lowest values). Its overall rank value across the five performance metrics is 1.8, much lower than the second position RA*k*EL, whose overall rank value is 2.375. This is also supported by its ranks on each of the eight datasets, from Table 2 to 9. The average performance over the eight datasets shown in Table 13 also supports this. Table 13 shows that MLKNN achieves the lowest values on the lowest-best metrics: Hamming Loss (HL), Coverage (CV), and Ranking Loss (RL), and achieves the highest value on the highest-best metric Average Precision (AP). Although we did not include the experimental results of three datasets (Cal500, Corel5k, and Bookmarks) into the summarization, their experimental results shown in Tables 10 through 12 completely support the conclusion.

Table 14 shows that RA*K*EL occupies the second position on all performance metrics, including the tied best in Hamming Loss (HL). Its overall rank 2.375 is also in the second position. Thus, we can conclude that RA*k*EL is the second best among the six algorithms. This is supported by its ranks on each of the eight datasets, from Table 2 to 9. The average performance over the eight datasets shown in Table 13 also supports this. Table 13 shows that RA*k*EL achieves the lowest value on the lowest-best metric One-Error (OE). It also achieves the second lowest in the lowest-best metrics: Hamming Loss (HL) and Ranking Loss (RL), and achieves the second highest value on the highest-best metric Average Precision (AP).

Table 14 shows that Adaboost.MH (AD) performs the worst. Its average ranks on all five performance metrics are the worst (highest values). Its overall rank value across the five performance metrics is 5.025, close to the maximum value 6.0. This is supported by its ranks on each of the eight datasets, from Table 2 to 9. The average performance over the eight datasets shown in Table 13 also supports this. Table 13 shows that AD achieves the highest values on the lowest-best metrics: One-Error (OE), Coverage (CV), and Ranking Loss (RL), and achieves the lowest value on the highest-best metric Average Precision (AP).

Table 14 shows that Classifier Chain (CC), Binary Relevance (BR), and Pruned Problem Transformation (PPT) take the middle positions. We can see that CC is the best, followed by BR, followed by PPT, among the three algorithms. Table 13 also shows this relationship. CC combines the binary relevance and the multi-label dependency. The better performance from CC shows that the label dependency exists, and needs to be utilized in multi-label classifications. That RA*k*EL performs the second also supports this. The potential drawback of Binary Relevance is the multi-label independency assumption.

In general, MLKNN performs the best, followed by RA*k*EL, followed by Classifier Chain and Binary Relevance. Classifier Chain improves the performance of Binary Relevance. Adaboost.MH performs the worst, followed by Pruned Problem Transformation. Why does Adaboost.MH perform the worst? We conjecture that the possible reason is its default base learner, decision stump. In the future, we will further investigate this.

5 Conclusion

In this paper, we provide an empirical comparison on six multi-label classification algorithms using eleven datasets. Our experiments show that the adaptive multi-label learning algorithm MLKNN performs the best, followed by RA*k*EL, followed by Classifier Chain and Binary Relevance. Adaboost.MH performs the worst, followed by Pruned Problem Transformation. This provides the guide for multi-label classification practitioners and saves their time to try and to estimate the possible achievement. This also stimulates us to study adapting traditional single label classification algorithms for multi-label problems. Our experimental results also provide us the confidence on the conjecture: there exist correlations among multi-labels. The multi-label independency assumption is not succeeded in most of datasets. How to utilize the correlations among these labels will shed a light for our future research.

We will continue to evaluate the performance of existing multi-label classification algorithms. In the same time, we are going to design novel algorithms for multi-classifications with the insights found in the experiments.

Acknowledgement. We thank the anonymous reviewers for the valuable comments. The work was supported by the National Science Foundation (IIS-1115417).

References

1. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining Multi-label Data. In: Maimon, O., Rokach, L. (eds.) Data Mining and Knowledge Discovery Handbook, 2nd edn. Springer (2010)
2. Klimt, B., Yang, Y.: Introducing the Enron corpus. In: First Conference on Email and Anti-Spam, CEAS (2004)
3. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. Machine Learning 85(3), 333–359 (2011)
4. Tsoumakas, G., Ioannis, K.: Multi-label Classification: An Overview. International Journal of Data Warehousing and Mining (2007)
5. Min-Ling, Z., Zhou, Z.: ML-KNN: A lazy learning approach to multi-label learning. Pattern Recognition 40(7), 2038–2048 (2007)
6. Arunadevi, J., Rajamani, V.: An Evolutionary Multi Label Classification Using Associative Rule Mining. International Journal of Soft Computing 6(2), 20–25 (2011)
7. Tsoumakas, G., Katakis, I., Vlahavas, I.: Random k-labelsets for multilabel classification. IEEE Transactions on Knowledge and Data Engineering 23(7), 1079–1089 (2011)
8. Yu-Yin, S., Zhang, Y., Zhi-Hua, Z.: Multi-label learning with weak label. In: Twenty-Fourth AAAI Conference on Artificial Intelligence (2010)
9. Alvares, C.E., Monard, M.C., Metz, J.: Multi-label Problem Transformation Methods: A Case Study. CLEI Electronic Journal 14(1), 4 (2011)
10. Tsoumakas, G., et al.: Mulan: A java library for multi-label learning. Journal of Machine Learning Research 1, 1–48 (2010)
11. Tao, L., Zhang, C., Zhu, S.: Empirical studies on multi-label classification. In: The Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2006 (2006)
12. Huang, D.-S., McGinnity, M., Heutte, L., Zhang, X.-P. (eds.): ICIC 2010. CCIS, vol. 93. Springer, Heidelberg (2010)
13. Colin, C., Mohammad, S.M., de Bruijn, B.: Binary classifiers and latent sequence models for emotion detection in suicide notes. Biomedical Informatics Insights 5(suppl. 1), 147 (2012)
14. Jesse, R.: A pruned problem transformation method for multi-label classification. In: Proc. 2008 New Zealand Computer Science Research Student Conference, NZCSRS 2008 (2008)

15. Yoav, F., Schapire, R., Abe, N.: A short introduction to boosting. Journal-Japanese Society for Artificial Intelligence 14, 771–780 (1999): 1612
16. Min-Ling, Z., Zhang, K.: Multi-label learning by exploiting label dependency. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (2010)
17. Schapire, R.E., Singer, Y.: Boostexter: A boosting-based system for text categorization. Machine Learning 39(2/3), 135–168 (2000)
18. Tsoumakas, G., Zhang, M.-L., Zhou, Z.-H.: Tutorial on learning from multi-label data. In: ECML/PKDD 2009, Bled, Slovenia (2009), http://www.ecmlpkdd2009.net/wp-content/uploads/2009/08/learningfrom-multi-label-data.pdf
19. Kumar, V., Wu, X.: Adaboost, The top ten algorithms in data mining, ch. 7, pp. 127–144. CRC Press (2009)